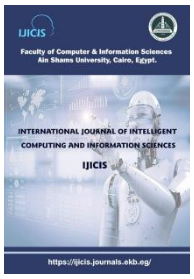
IJICIS, Vol.24, No.3, 41-57
DOI :10.21608/ijicis.2024.317982.1352

International Journal of Intelligent Computing and Information Sciences
https://ijicis.journals.ekb.eg/


IJICIS
Faculty of Computer & Information Sciences Ain Shams University, Cairo, Egypt.
INTERNATIONAL JOURNAL OF INTELLIGENT COMPUTING AND INFORMATION SCIENCES
IJICIS
https://ijicis.journals.ekb.eg/

STUDYING THE IMPACT OF DATASET BALANCING ON MACHINE LEARNING-BASED INTRUSION DETECTION SYSTEMS FOR IOT

Salma Abdelhamid*

Computer Science Department,
Faculty of Computers and Information Technology,
Future University in Egypt,
Cairo, Egypt
salma.abdelhamed@fue.edu.eg

Islam Hegazy

Computer Science Department,
Faculty of Computer and Information Sciences, Ain Shams University,
Cairo, Egypt
islheg@cis.asu.edu.eg

Mostafa Aref

Computer Science Department,
Faculty of Computer and Information Sciences, Ain Shams University,
Cairo, Egypt
mostafa.aref@cis.asu.edu.eg

Mohamed Roushdy

Computer Science Department,
Faculty of Computer Science and Information Technology, Innovation University,
Al-Sharqia, Egypt
mohamed.roushdy@iu.edu.eg

Received 2024-09-03; Revised 2024-09-03; Accepted 2024-09-11

Abstract: *Internet of Things (IoT) networks are integral to modern life due to their pervasive connectivity and automation capabilities. Intrusion Detection Systems (IDS) are crucial in IoT ecosystems to countermeasure attacks that can compromise devices and disrupt essential services. Their role is vital in maintaining the integrity, confidentiality, and availability of data within these networks. The effectiveness of these security systems is fundamentally dependent on the robustness of learning algorithms and the quality of the datasets utilized. Class imbalance is a common challenge in real-world datasets, where certain classes are represented by significantly fewer instances compared to others. This paper studies the impact of balancing the BoT-IoT dataset on the performance of Machine Learning (ML) based IDSs using three algorithms: K-Nearest Neighbors (KNN), Gradient Boosting (GB), and Support Vector Machine (SVM). We apply two resampling techniques: random upsampling and Synthetic Minority Over-sampling Technique (SMOTE). The results show that dataset balancing improves F1-scores across all the algorithms. Minority classes F1-scores increase in KNN, GB, and SVM from 0.77 to 1, 0 to 0.989, and 0 to 0.999; respectively. Our findings prove that balanced datasets lead to more dependable and robust IDSs that are capable of handling real-world data with varied class distributions.*

Keywords: *Internet of Things, Intrusion Detection, Machine Learning, SMOTE.*

1. Introduction

***Corresponding Author**: Salma Abdelhamid

Computer Science Department, Faculty of Computers and Information Technology, Future University in Egypt, Cairo, Egypt

Email address: salma.abdelhamed@fue.edu.eg

The Internet of Things (IoT) is a vast network of interconnected exchange data and information in both wired and wireless modes to accomplish certain tasks or run specific applications. IoT devices are used in various domains such as smart cities, education, fleet management, and industrial plants [1, 2]. They are even used as personal cardiac monitoring wristlet [3]. Nevertheless, IoT gadgets are low-powered vulnerable devices that attract intruders to exploit them to gain access to critical data or services in the system. Therefore, in our contemporary connected world, Intrusion Detection Systems (IDS) are essential aspects of securing IoT networks [4]. An IDS is a combination of software and hardware components that aim to detect any malicious behavior within the network. Over the past years, researchers have been exploring the power of Machine Learning (ML) algorithms to provide an efficient IoT-tailored IDS. In contrast to conventional firewalls and detection methods, ML is capable of handling large data [5]. The intelligence of these techniques and their ability to learn are used to monitor the system and recognize any abnormal behavior.
Network security researchers have created intrusion datasets to aid the IDSs in identifying diverse attacks. In the context of these publicly available datasets, researchers build methodologies and tools for building an efficient security system. One of the prevalent issues recognized in numerous scholarly articles is that specific attack categories occur infrequently, whereas other types of attacks constitute the largest part of the dataset. This imbalance downgrades the performance of the system, as the model succeeds at identifying the majority classes but struggles to identify minority classes [6, 7]. Dataset balancing techniques involve methods such as oversampling, where minority class instances are augmented, and undersampling, where majority class instances are reduced. Our study aims to provide valuable insights into the significance of dataset balance and its implications for intrusion detection efficacy using prominent ML algorithms. We train and evaluate the models using BoT-IoT dataset [8] which encloses four attacks. These attacks are Denial of Service (DoS), Distributed Denial of Service (DDoS), Reconnaissance, and theft attacks. The key contributions of this research are:

- Addressing the problem of imbalanced datasets using data resampling techniques. We apply Synthetic Minority Oversampling Technique (SMOTE) and random under-sampling technique to the dataset to elevate the performance of the models.
- Conducting an empirical analysis and comparing the performance of machine learning models on balanced and unbalanced datasets. We provide a thorough view of the impact of dataset balance on classification performance by systematically evaluating three different algorithms with different hyperparameters. The ML algorithms are K-Nearest Neighbors (KNN), Gradient Boosting (GB), and Support Vector Machines (SVM).
- Assessing the performance of ML classification models using different evaluation metrics, including accuracy, precision, recall, and F1-score. This allows for a better understanding of each model’s performance in the presence of class imbalance.

The rest of the paper is organized as follows. section II gives insights into the background and the related work in this domain. In section III, we illustrate the proposed methodology along with the data preprocessing phases. Section IV demonstrates the experimental work and findings. Lastly, Section V discloses our study's conclusion and future directions.

2. Background and Related Work

2.1. Intrusion Detection Systems

In the realm of IoT, and with the continuous growth of the embedding of vulnerable IoT devices, we cannot overstate the importance of creating an efficient IDS. The primary function of IDS is to detect abnormal activities and unauthorized entries into IoT networks or systems. IoT environments are at risk of a vast array of attacks, and without enough security measures, the system and devices are left open to potential threats. By employing IDSs, network traffic can be analyzed, potential threats can be identified, and administrators can be promptly notified of any dubious activities. As illustrated in Figure 1, the general framework of an IDS comprises numerous stages. The primary processes are data collecting, data pre-processing, intrusion detection, and alarm activation.
IDSs are classified based upon information source, detection method, placement strategy, and response mechanism. Based on the information source, IDSs are categorized as Host-based (HIDS), Network-based (NIDS), or Hybrid. HIDS monitor activities on a single host, analyzing operating system data but consuming host resources, whereas NIDS observe network traffic across the entire network, and Hybrid IDSs combine both for improved accuracy. Based on detection methods, IDSs use Signature (Misuse) Detection, which relies on known attack signatures but cannot detect new threats, or Anomaly Detection, identifies abnormal behaviors but produces false positives [9]. Specification-based IDSs define normal behavior to detect anomalies, while Hybrid systems merge multiple techniques for greater robustness. Placement strategies include Centralized, where a single node monitors the network, Distributed, where devices monitor themselves, and Hybrid, which combines both for increased reliability. Finally, based on response mechanisms, IDSs can be active, automatically responding to threats in real-time, or passive, alerting administrators to respond manually.

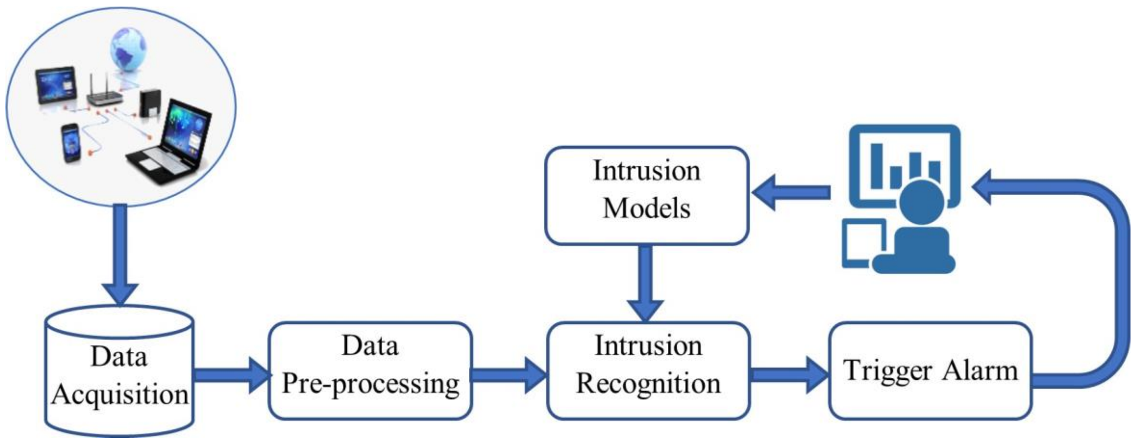


Figure 1: The general framework of an IDS

2.2. Intrusion Detection Datasets

To evaluate intrusion detection systems, it is crucial to use dependable datasets that enclose both legitimate and malicious data. Furthermore, the comparison of diverse proposed studies demands having benchmark datasets. Below is a brief description of some of the benchmark intrusion detection datasets.

- **DARPA98** [10]: The researchers in the Defense Advanced Research Project Agency (DARPA) evaluation program created this dataset in 1998. It originated from audit logs and network traffic, and it is composed of two subsets. The first one comprises seven weeks of network-based attacks with regular data and is used for training the models. The second one comprises two weeks of attacks mixed in with regular data and is used for testing the model. However, this is an old dataset that does not fully reflect the variety and severity of contemporary cyber threats.

- **KDD Cup 99 / KDD99** [11]: This dataset was utilized in the Knowledge Discovery and Data Mining Tools (KDD) Competition. It is an improved version of the previously released DARPA dataset, where the network traces were converted into 41 connection characteristics based on Transmission Control Protocol (TCP) connections. In the creation of this dataset, Lincoln Labs researchers simulated a typical US Air Force Local Area Network (LAN) using a variety of integrated intrusion attack techniques. Yet, the dataset does not fully represent the complexity of the existing cyber-attacks, and it is extremely imbalanced.
- **NSL-KDD** [12]**:** The NSL-KDD dataset is a widely used benchmark dataset for intrusion detection research. It is an improved version of the original KDD Cup 99 dataset, addressing some of its limitations such as redundancy and irrelevant features. With its balanced distribution of attack types and reduced feature space, NSL-KDD facilitates more accurate and robust evaluations of intrusion detection systems. It has 125,973 records for training and 22,544 records for testing. It encloses 22 intrusion attacks and 41 attributes. The drawback of this dataset is that it is synthetic and does not fully represent the complexities and diversities of real-world network traffic.
- **Kyoto 2006+** [13]: This dataset was collected over three years, from 2006 to 2009 using real network data collected through honeypots and servers of Kyoto University. It includes 14 conventional features originating from the KDD99 dataset, as well as 10 additional extracted features.
- **UNSW-NB15** [14]: The primary purpose of creating this dataset was to combine traditional normal activities with simulated contemporary harmful behaviors. It includes various attack scenarios. Nevertheless, this dataset does not cover some types of intrusion detection, such as host-based or application-layer intrusion detection. In addition to that, there is insufficient variation in the normal network traffic which could affect the classification accuracy of the IDS.
- **CICIDS 2017** [15]: This dataset contains a broad range of the most prevalent attacks derived from real network traffic features. It contains both benign traffic and attack instances in their PCAP files. The dataset was created at the Canadian Institute of Cyber Security and the authors used the CICFlowMeter to analyze packet captures and export the dataset's main features.
- **Bot-IoT:** The BoT-IoT dataset consists of real-world IoT network traffic captured from a smart campus environment at UNSW Canberra cyber center. It includes both benign and malicious activities. It encloses more than 72 million records with attacks such as DoS, DDoS, Reconnaissance, and theft attacks. This dataset provides a realistic testbed and organizes collected traffic depending on attack categories.

In our study, we use the BoT-IoT dataset for training and testing our models. Unlike other intrusion detection datasets, the BoT-IoT dataset offers a unique advantage of capturing real-world IoT network traffic from a smart environment. This provides a highly realistic and diverse set of scenarios for evaluation. Additionally, the dataset includes both benign and malicious activities, allowing for the development and assessment of intrusion detection systems under various threat scenarios. Nevertheless, Intrusion detection state-of-the-art benchmark datasets include class imbalances, and this problem should be addressed in the data preprocessing phase. Table 1 summarizes the previously mentioned datasets.

Table 1 Summary of Intrusion Detection Datasets

Dataset	Year	Developer	No. of features	No. of records	Attacks	Limitation
DARPA98	1998	MIT Lincoln Laboratory	41	409020	DoS, R2L, U2R, and Probing	Outdated attacks Not real traffic Lack of false positives
KDD99	1999	University of California	41	4,898,431	DoS, R2L, U2R, and Probing	Outdated attacks. Extremely unbalanced.
NSL-KDD	2009	University of California	41	148,517	DoS, R2L, U2R, and Probing	Imbalanced Limited attack types.
Kyoto 2006+	2006	Kyoto University	24	93,076,270	Normal and Attack sessions	Unrealistic normal traffic. Lack of attack details.
UNSW-NB15	2015	University of New South Wales	49	2,540,044	Analysis, Backdoor, DoS, Exploits, Fuzzers, Generic, Shellcode, Worms, and Reconnaissance	Imbalanced Lack of some intrusions Unrealistic IoT nature Insufficient normal traffic
CICIDS2017	2017	Canadian Institute of Cyber Security	79	2,830,743	Botnet, Brute Force, Web, DDoS, DoS, Heartbleed, Infiltration, and Port Scan.	Imbalanced Includes synthetic traffic. Feature Redundancy Limited Documentation
Bot-IoT	2018	University of New South Wales	43	73,370,443	DDoS, DoS, Theft, and Reconnaissance	Highly Imbalanced Large in size Lack of Longitudinal data

2.3. Literature Review

Machine learning models are being used increasingly to detect security risks and vulnerabilities in IoT environments. Machine learning has effectively performed in spam, fraud, and anomaly detection. Security researchers have proposed using ML models in IDSs as a viable option for designing anomaly-based IDS, as evidenced by the accuracy and effectiveness of the proposed techniques. This subsection overviews previously proposed ML-based IDSs in IoT.

Gao et al. [16] developed an IDS based on an ensemble adaptive voting technique. They applied four distinct algorithms: Decision Tree (DT), Random Forest (RF), KNN, and Deep Neural Networks. They verified their technique using an NSL-KDD-Test+ file. The DT's accuracy is 84.2%, while the adaptive algorithm's accuracy is 85.2%. The authors stated that their ensemble adaptive model enhances detection accuracy. However, the model performed poorly with minor classes of the dataset.
Chkirbene et al. [17] introduced a hybrid technique that incorporates two ML algorithms. They employed RF to identify the important features, then they used Classification and Regression Trees (CART) to categorize the various assault types. They tested their system using the UNSW-NB15 dataset. The accuracy for the UNSW-NB15 and KDD99 datasets were 95.73% and 97.03%, respectively.
Ferrag et al. [18] introduced a Rules and Decision Tree-Based Intrusion Detection System (RDTIDS) for IoT Networks. They employed many techniques including REP Tree, JRip algorithm, and Forest PA. The system employed three classifiers, with the third taking into account the results of the previous two. The model achieved greater than 96% accuracy when evaluated against real traffic data sets CIC-IDS-2017 and BoT-IoT.

Dina et al. [19] presented an intrusion detection system based on decision trees. They classified attacks as normal, dos, probe, r2l, or u2r. When evaluated on the KDDTest, their DT algorithm obtained an accuracy of 75.22%, compared to 73.26% for SVM and 62.73% for RF.
Douiba et al. [20] used Gradient Boosting (GB) and Decision Tree (DT) algorithms to build a refined IDS. They tested and validated their models using NSL-KDD, IoT-23, BoT-IoT, and Edge-IIoT datasets. The model had good overall performance metrics.
Xu et al. [21] created an automated method for detecting 22 IoT attacks using machine learning and an edge server. Their model aimed to classify network intrusions based on characteristics such as server count, error rate, and network protocol. They used the SMOTE technique to balance the KDDcup99 dataset. Their experimental works showed that the proposed model outperformed prior models, with high classification accuracy.

ML-based IDSs have been developed over the past years, with researchers focusing on improving detection accuracy. However, relying solely on accuracy as the evaluation metric can be misleading, as it fails to account for class imbalances, where some attack types are vastly outnumbered by normal instances. Therefore, checking per-class performance is crucial as it provides insights into the detection capabilities for each attack type, highlighting potential weaknesses or biases in the system. By analyzing per-class performance metrics such as precision, recall, and F1-score, we can better understand the system's ability to detect specific attacks accurately, enabling more effective threat mitigation strategies.

3. Proposed Methodology

This research examines the effect of data balancing on the classification of different ML algorithms. The general research framework is shown in Figure 2.

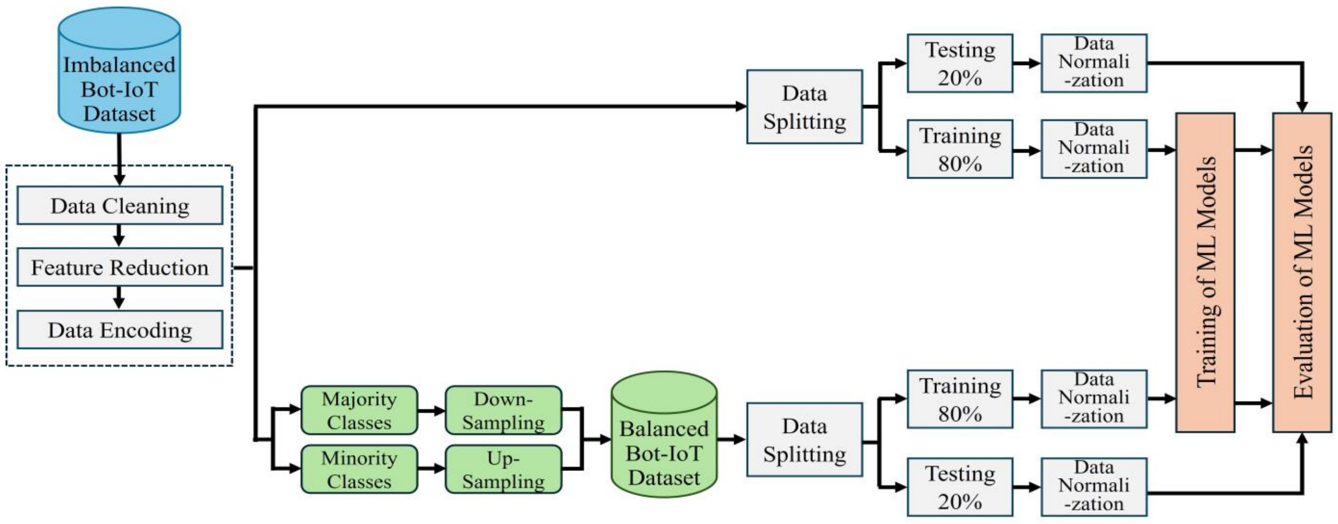


Figure 2: The general framework of the proposed system

3.1. Dataset Preprocessing

We train and evaluate the implemented IDSs using the Bot-IoT benchmark dataset, which has 43 features, allowing comprehensive analysis and ensuring flexibility in feature reproducibility. The full feature set includes features that are significant in certain contexts or for specific types of analysis. Nevertheless, we use the 10-Best subset. This subset holds ten features with the best correlation coefficient and joint entropy. Using the 10-Best subset eliminates the need for extensive feature engineering while preserving the main discriminative features essential for intrusion detection and saving resources during the models' development. The authors additionally enclosed 5 independent flow identifiers and 3 dependent features. The flow identifiers are "saddr," "sport," "daddr," "dport," and "proto." The 3 dependent features are "attack," "category," and "subcategory." Moreover, "pkSeqID" is a numerical feature that serves as a row identifier. Table 2 shows the description and datatype of the 10-Best subset features.

Table 2: The Description of the 10-Best BoT-IoT subset features

Features	**Description**	**Data Type**
pkSeqID	Row identifier	int64
proto	Textual representation of transaction protocols in network flow	object
saddr	Source IP address	object
sport	Source port number	object
dadrr	Destination IP address	object
dport	Destination IP address	object
seq	Argus sequence number	int64
stddev	Standard deviation of aggregated records	float64
N_IN_Conn_P_SrcIP	Number of inbound connections per source IP	int64
min	Minimum duration of aggregated records	float64
state_number	Numerical representation of feature state	int64
mean	Average duration of aggregated records	float64
N_IN_Conn_P_DstIP	Number of inbound connections per destination IP	int64
drate	Destination-to-source packets per second	float64
srate	Source-to-destination packets per second	float64
max	Maximum duration of aggregated records	float64
attack	Numerical representation of instance nature (0= Normal, 1=attack)	int64
category	Attack category	object
subcategory	Attack subcategory	object

Dataset preprocessing is a necessary stage after its acquisition. The applied preprocessing steps are:

- **Data Cleaning:** During the preliminary stages of data preprocessing, we remove all null, replicate, and infinite values from the dataset.
- **Feature reduction:** We reduce the number of features by dropping the flow identifiers, as they convey localized insights that cannot be broadened, and might lead to skewed predictions. We drop the "pkSeqID" column as it is a numerical index that does not provide significant information. Furthermore, we drop the "subcategory" feature. The "attack" feature, which numerically represents the attack category, is dropped because this information is already conveyed in the "category" feature.
- **Data Encoding:** We change the categorical features to numerical representations suitable for machine learning algorithms. In our study we consider five classes in the BoT-IoT dataset: DoS, DDoS, Normal, Reconnaissance, and Theft. Therefore, we encode a unique number for each of these five classes.
- **Data Normalization:** We use the MinMax Scalar to bring all the features to a similar scale to prevent certain features from dominating others. This transformation preserves the shape of the

original distribution but within certain bounds; from 0 to 1. We apply the data normalization after splitting the dataset into training and testing subsets to avoid test-to-train leakage for a more robust IDS [22]. The new normalized feature value (X_{new}), of a feature (X) is calculated as follows:

$$X_{i\,new} = \frac{X - X_{min}}{X_{max} - X_{min}} \qquad (1)$$

Where X_{min} and X_{max} are the minimum and maximum values of X, respectively.

- **Data Splitting:** We split our dataset into 80% for training and 20% for testing. This process helps to evaluate the performance of models on unseen data, ensuring robustness and generalization.

3.2. Dataset Balancing

The BoT-IoT is highly imbalanced. To reach a balanced dataset, we apply two resampling techniques:

- **Random Down-sampling:** This technique is used to address the dataset imbalance when the majority class significantly outnumbers the minority class. It randomly selects instances from the majority class and removes them from the dataset for a more balanced class distribution.
- **Synthetic Minority Oversampling Technique (SMOTE)** [23]**:** SMOTE is an up-sampling technique that generates novel data points based on the k-nearest clustering-based algorithm. SMOTE holds greater advantage than random up-sampling, as it creates new points rather than simply duplicating already existing ones. It creates for each minority point x_i, a new point x'_i, that depends on the randomly selected x_j from x_i's nearest (k) neighbors, and λ is a random value ranged in [0, 1]. The functionality of this up-sampling technique is defined in Eq. (2).

$$x'_i = x'_i + \lambda(x'_j - x'_i) \qquad (2)$$

3.3. Machine Learning Classification Techniques

Machine learning is a branch of AI in which computers aim to enhance their performance through learning and experience gaining. It comprises the development of statistical algorithms to extract specific patterns and make decisions or predictions. ML techniques are primarily classified as unsupervised or supervised learning, with other classifications including semi-supervised and reinforced learning. In supervised approaches, the model is trained on labeled datasets, whereas unsupervised models take unlabeled input data and attempt to infer patterns and features on their own. This subsection gives a brief overview of the ML algorithms implemented in this study.

- **K-Nearest Neighbor (KNN):** This is a fundamental ML algorithm that is used for classification and regression tasks. In KNN, the classification of a datapoint is determined by the majority class or average value of its nearest neighbors in the feature space. The number of neighbors considered in the classification is determined by the user-defined constant, "k." Larger values of K smooth out noises but potentially cause overfitting, while smaller values of K can capture more details but might be more susceptible to noise. Despite being an adaptive algorithm that effectively manages newly acquired data, KNN performs badly when the class distribution is skewed.
- **Gradient Boosting:** Gradient boosting is an ensemble learning strategy that creates its prediction model by integrating multiple weak learners. It applies boosting techniques to solve regression and classification problems. Typically, the weak learners in this algorithm are Decision Trees (DTs). In each iteration, a new decision tree is trained to forecast the residuals of the preceding model, and the overall prediction is updated based on the predictions from all trees.

- **Support Vector Machine (SVM)**: SVM is a supervised method used for classification and regression. It translates data into higher dimensions and determines the maximum margin hyperplane that distinguishes between normal and abnormal cases. Various separation hyper-planes are obtained via a Kernel, which can be linear, polynomial, hyperbolic tangent, or Gaussian Radial Basis Function. This algorithm is memory-efficient and can detect real-time attacks by analyzing attack patterns while training the data. However, SVM is impractical with large datasets, and it is sensitive to any data noise near the hyperplane.

We intentionally selected these three algorithms because they represent diverse learning paradigms. The instance-based KNN, ensemble-based GB, and margin-based SVM provide perspectives on how dataset balancing impacts model performance. Moreover, these algorithms are widely adopted in different fields, making them reliable benchmarks. Their inclusion ensures that the study covers a broad spectrum of machine learning techniques, offering insights that are robust and generalizable across various IDS scenarios in IoT environments.

4. Experimental Work and Discussion

4.1. Environmental Setup and Hyperparameters

We implement our models via the online Google Colaboratory (Colab), provided with Intel(R) Xeon vCPU @2.3 GHz, 2 cores, and 13 GB RAM. We use Python 3.1 and Keras 2.12 API running on Tensorflow 2.13 as the backend. Our selected models have main hyperparameters. "k" in KNN is the number of nearest neighbors used for classification, "C" in SVM is the regularization parameter that controls the trade-off between maximizing the margin and minimizing the classification error, and the number of estimators in GB refers to the number of trees used. We examine different values of these hyperparameters to assess their impact on model performance.

4.2. Evaluation Metrics

We measure the functionality and efficiency of a learning-based model through certain evaluation metrics. These metrics assist in comparing different models or algorithms and offer insight into how well the model is working. The classification predictions fall under four categories: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). TP and TN are when the predictions are correct, and they belong to the positive and negative classes, respectively. FP refers to an incorrect prediction of the positive class. While FN means that the model predicted a negative class which is an incorrect prediction. The combination of these measures results in our main evaluation metrics, which are accuracy, precision, recall, and F1-score. These metrics are calculated as follows:

$$\text{Accuracy} = \frac{\text{No. of correct predictions}}{\text{Total predictions}} = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

$$\text{Precision} = \frac{\text{No. of correct positive predictions}}{\text{Total positive predictions}} = \frac{TP}{TP + FP} \tag{4}$$

$$\text{Recall} = \frac{\text{No. of correct positive predictions}}{\text{Total No. of positive instances}} = \frac{TP}{TP + FN} \tag{5}$$

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{6}$$

For the averaged performance metrics of each model, we study the macro average results, which is the arithmetic mean of the result across all classes. This approach is better in studying the effect of imbalanced datasets as it treats all classes equally, ensuring that the performance on minority classes is not overshadowed by the majority class. We also examine the confusion matrix because it provides detailed insights into the performance of our classification model and the types of errors it is making.

4.3. Addressing the Data Imbalance

As previously mentioned, the BoT-IoT dataset is extremely imbalanced, and we address this problem by under-sampling and oversampling techniques. For both techniques, we select the "Reconnaissance" class as our reference class for resampling because it represents the average class distribution in the dataset. This allows us to adjust the size of other classes in relation to this central class, either by upsampling majority classes or downsampling minority classes, to achieve a more balanced dataset. We randomly under-sample the “DoS” and “DDoS.” We up-sample the minority classes, “Normal” and “Theft” using SMOTE with k=2. The distribution percentage of the classes before and after the balancing techniques is shown in Figure 3. As seen in Figure (3-a), the BoT-IoT dataset is extremely imbalanced that the “Theft” and “Normal” classes cannot be seen in the pie chart distribution.

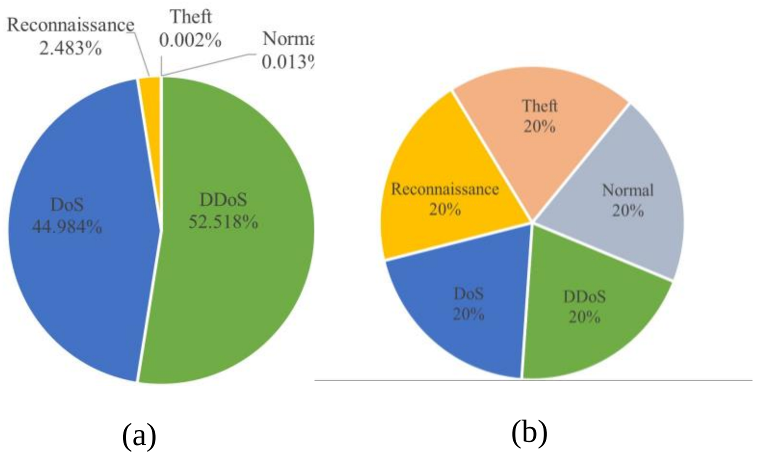


Figure 3: The distribution percentage of the 10-Best BoT-IoT dataset before and after resampling. (a) Distribution before resampling, illustrating the original imbalanced data distribution. (b) Distribution after resampling, showing the balanced data distribution.

4.4. Results and Discussions

Despite the high accuracies in some experiments, these high results are misleading as they reflect the models’ abilities to classify the majority class correctly. Therefore, for legitimate evaluations, we must consider the models' precision, recall, and F1-score. The analysis and discussion of each model is presented in the following.

1. **K-Nearest Neighbors:** To observe how model complexity affects performance, we implement the KNN model using 5, 9, and 13 neighbors, larger values smoothing out noise and smaller values capturing finer details. The classification results of the KNN models are presented in Table 4.

- **Performance with imbalanced data:** Table 4 illustrates that the model's performance on the imbalanced dataset highlights disparities between classes. With K=5, the model achieves high results for the DoS and DDoS. However, the Theft class has a precision of 0.929 but a recall of only 0.813, leading to a low F1-score of 0.8667. The Normal class has a precision of 0.974, a recall of 0.8929, and an F1-score of 0.9317. The macro average F1-score is lower than the individual majority class scores which demonstrates the model's bias towards majority classes.
- **Performance with balanced data:** The model achieves high results for the balanced dataset across all classes. The minority classes achieve scores that range from 0.99 to 1. The macro average F1-scores for the balanced dataset increases, reflecting a more reliable and unbiased detection capability. These enhancements highlight the importance of balancing datasets in intrusion detection for IoT environments, leading to more robust and comprehensive security measures. Despite the lower accuracies in the balanced datasets, the metrics for precision, recall, and F1-score are more consistent and higher. This reflects a more reliable and fair evaluation of the model's performance across all classes. The balanced dataset reduces the skewness caused by imbalanced class distributions, leading to a model that performs well across the board and provides more meaningful performance metrics.

Figure 4 shows the confusion matrices of KNN models. The top row shows results from the imbalanced dataset with varying values of k (5, 9, and 13), while the bottom row reflects the balanced dataset. The matrices reveal that as k increases, the classification accuracy for the imbalanced dataset degrades slightly, especially for minority classes like Normal and Theft. In contrast, balancing the dataset leads to more consistent and accurate predictions across all categories, even as k changes. This demonstrates the importance of dataset balancing in improving KNN's performance, particularly in handling minority classes more effectively.

Table 4: The classification results of the KNN models.

Model		**Class**	**Imbalanced Dataset**				**Balanced Dataset**			
			Precision	**Recall**	**F1-score**	**Accuracy**	**Precision**	**Recall**	**F1-score**	**Accuracy**
k-Nearest Neighbors	**k = 5**	DoS	0.994	0.994	0.994	99.5%	0.979	0.983	0.981	99.2%
		DDoS	0.995	0.995	0.995		0.984	0.980	0.982	
		Reconnaissance	0.997	0.992	0.995		0.996	0.996	0.996	
		Theft	0.929	0.813	0.867		1	1	1	
		Normal	0.974	0.893	0.932		0.999	1	0.999	
		Macro Average	0.978	0.937	0.956		0.992	0.992	0.992	
	k = 9	DoS	0.992	0.992	0.992	99.2%	0.976	0.980	0.978	99.0%
		DDoS	0.993	0.993	0.993		0.982	0.975	0.978	
		Reconnaissance	0.993	0.988	0.990		0.995	0.995	0.995	
		Theft	0.722	0.812	0.765		1	1	1	
		Normal	0.941	0.762	0.842		0.999	1	0.999	
		Macro Average	0.928	0.909	0.916		0.990	0.990	0.990	
	k = 13	DoS	0.990	0.991	0.990	99.1%	0.973	0.981	0.977	98.9%
		DDoS	0.992	0.992	0.992		0.983	0.971	0.977	
		Reconnaissance	0.990	0.984	0.987		0.993	0.995	0.994	
		Theft	0.722	0.812	0.765		0.999	1	1	
		Normal	0.948	0.655	0.775		0.998	1	0.999	
		Macro Average	0.928	0.887	0.902		0.989	0.989	0.989	

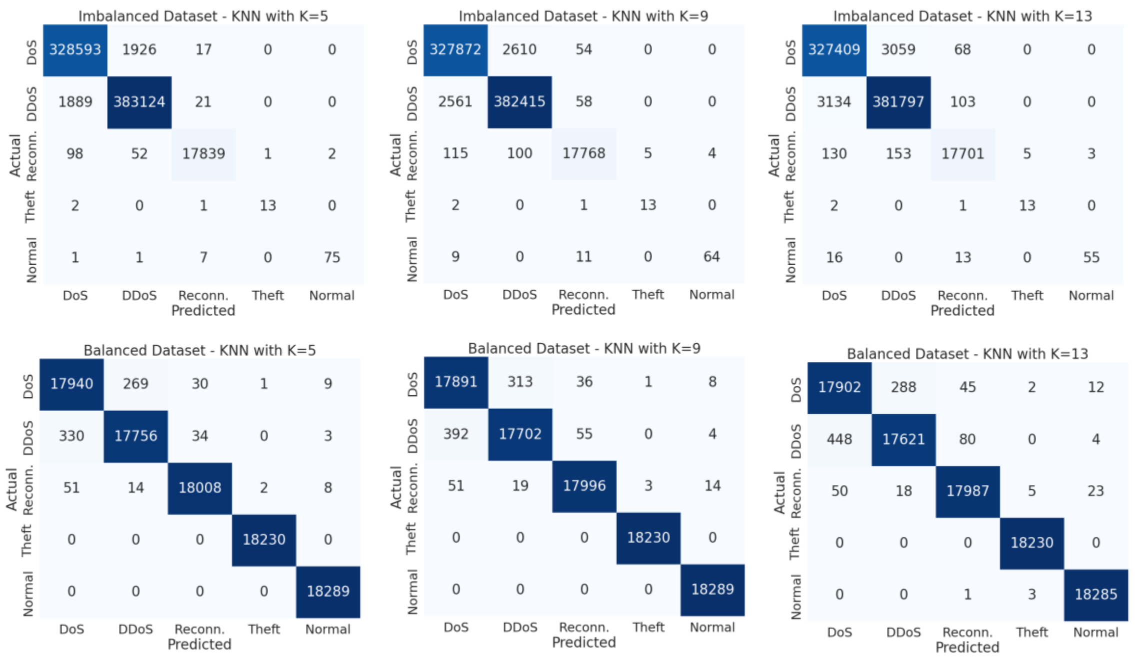


Figure 4: The confusion matrices of KNN models with balanced and imbalanced datasets.

2. **Gradient Boosting:** We conducted experiments for the GB using 10, 50, and 100 estimators to explore how the number of trees impacts model accuracy and generalization. The results are shown in Table 5.

   - **Performance with imbalanced data:** With all the estimators, the model fails to detect the minority classes; “Theft” and “Reconnaissance” with zero results across all metrics. With 10 estimators, the model shows high precision for “DoS” and “Reconnaissance” attacks but significantly lower recall, indicating that it correctly identifies positives but misses many actual positives of these attacks. The “DDoS” attack has high recall but low precision, meaning it captures most positives but with many false positives. Despite the good accuracies, the models have low F1-scores which are: 0.338, 0.528, and 0.549 with 10, 50, and 100 estimators, respectively. This signifies that the model is performing poorly in balancing precision and recall, and the results include unsatisfactory false positives and false negatives.
   - **Performance with balanced data:** Balancing the dataset succeeds in significantly enhancing the precision and recall of the minority classes from 0 to reach ranges between 0.86 to 0.99 and 0.75 to 1 for precision and recall, respectively. With 10, 50, and 100 estimators respectively, the accuracy increases to approximately 80%, 92%, and 95%, and the F1-score remarkably incremented to 0.79, 0.92, and 0.95. These elevated results prove that balancing the datasets leads to more accurate classification and reduced false positives and negatives.

Figure 5 shows the GB confusion matrices. As illustrated, the GB algorithm suffers from significant bias when trained on an imbalanced dataset, consistently favoring majority classes like while neglecting minority classes. This skewness persists even as the number of estimators increases, highlighting the challenge of imbalanced data. In contrast, the balanced dataset leads to more accurate classification across all classes. As the number of estimators increases, the balanced dataset consistently improves the model's performance on minority classes, emphasizing the importance of addressing class imbalances for reliable and efficient results.

Table 5: The classification results of the GB models.

Model		Class	Imbalanced Dataset				Balanced Dataset			
			Precision	Recall	F1-score	Accuracy	Precision	Recall	F1-score	Accuracy
Gradient Boosting	**10 Estimators**	DoS	0.959	0.459	0.621	73.3%	0.808	0.553	0.656	79.50%
		DDoS	0.669	0.996	0.801		0.649	0.860	0.739	
		Reconnaissance	0.989	0.155	0.268		0.807	0.815	0.811	
		Theft	0	0	0		0.900	1	0.947	
		Normal	0	0	0		0.862	0.751	0.803	
		Macro Average	0.524	0.322	0.338		0.805	0.796	0.791	
	50 Estimators	DoS	0.854	0.890	0.872	88.1%	0.914	0.734	0.814	91.8%.
		DDoS	0.902	0.878	0.890		0.812	0.916	0.861	
		Reconnaissance	0.997	0.788	0.880		0.943	0.980	0.961	
		Theft	0	0	0		0.978	0.982	0.980	
		Normal	0	0	0		0.953	0.978	0.965	
		Macro Average	0.551	0.511	0.528		0.920	0.918	0.916	
	100 Estimators	DoS	0.917	0.864	0.890	90.3%	0.925	0.826	0.873	94.7%
		DDoS	0.888	0.937	0.912		0.883	0.925	0.904	
		Reconnaissance	0.996	0.895	0.943		0.955	0.989	0.972	
		Theft	0	0	0		0.993	1	0.996	
		Normal	0	0	0		0.980	0.997	0.989	
		Macro Average	0.560	0.539	0.549		0.947	0.947	0.947	

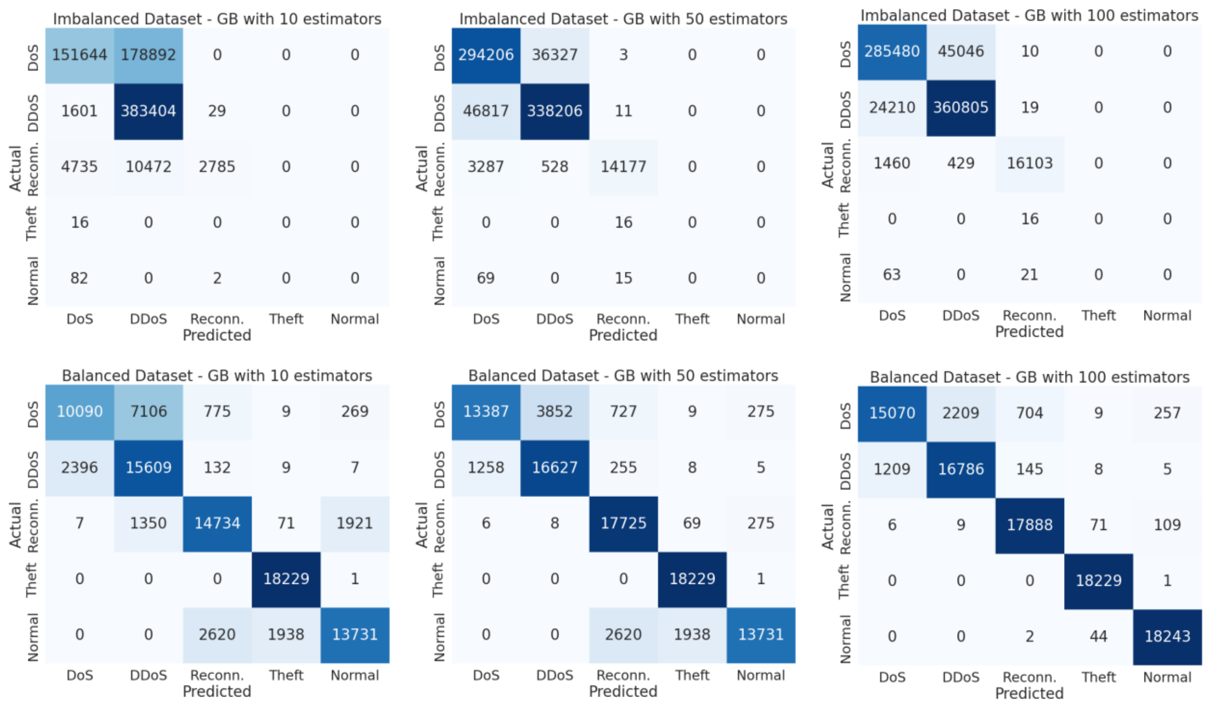


Figure 5: The confusion matrices of GB models using balanced and imbalanced datasets.

3. **Support Vector Machine**: The SVM models' performance on the imbalanced dataset shows significant variability across different classes and values of the regularization parameter. The results are recorded in Table 6. The regularization parameter "C" was tuned to balance the trade-off between maximizing the margin and minimizing classification errors, as higher values may lead to overfitting while lower values provide a more generalized model.

- **Performance with imbalanced data:** With C = 1 the model achieves a moderate precision of 0.675 and recall of 0.509 for the “DoS” class, resulting in an F1-score of 0.58. The “DDoS” class has an F1-score of 0.712. However, minority classes like “Theft” and “Normal” are not detected at all, reflected by zero values in their precision, recall, and F1-scores. Increasing C to 5 improves the metrics for most classes and boosts the total accuracy from 0.6638 to 0.9395. However, the performance degraded with C = 10, illustrating the complex relationship between the regularization parameter and model performance in imbalanced contexts. Nevertheless, the F1-score for the imbalanced dataset remains low, indicating the challenge of handling imbalanced data.
- **Performance with balanced data:** Balancing the dataset remarkably enhances the results across all metrics for most classes. It increases the F1-score of the minor classes to above 0.99 across all values of C, indicating an improvement in the models’ balance between precision and recall. This means that the model is more accurate in identifying positive instances while minimizing both false positives and false negatives. The results show that SVM is sensitive to imbalanced classes’ distribution and to the choice of its regularization parameter.

The confusion matrices for the SVM algorithm are displayed in Figure 6. Increasing the regulation parameter in the imbalanced dataset leads to better classification of majority, but it continues to perform poorly on minority. At C=10, the "DoS" class has 223,257 correct predictions, while the minority classes are still largely misclassified, reflecting severe skewness. In contrast, the balanced dataset results in more evenly distributed classifications across all classes, even though overall accuracy might slightly decrease. For example, at C=10, the "Reconnaissance" class achieves 17,865 correct predictions compared to just 17,265 in the imbalanced dataset, indicating more reliable performance. The balanced dataset reduces the bias towards majority classes, leading to more effective classification across all classes.

Table 6: The classification results of the SVM models.

Model		**Class**	**Imbalanced Dataset**				**Balanced Dataset**			
			Precision	**Recall**	**F1-score**	**Accuracy**	**Precision**	**Recall**	**F1-score**	**Accuracy**
Support Vector Machine	**C=1**	DoS	0.675	0.509	0.58	66.4%	0.924	0.956	0.94	96.8%
		DDoS	0.652	0.784	0.712		0.969	0.914	0.941	
		Reconnaissance	0.787	0.952	0.862		0.971	0.982	0.977	
		Theft	0	0	0		0.997	0.99	0.994	
		Normal	0	0	0		0.982	0.999	0.99	
		Macro Average	0.423	0.449	0.43078		0.969	0.968	0.96825	
	C=5	DoS	0.915	0.963	0.939	94%	0.935	0.967	0.95	97.6%
		DDoS	0.969	0.918	0.943		0.974	0.926	0.95	
		Reconnaissance	0.834	0.96	0.893		0.979	0.988	0.983	
		Theft	0	0	0		0.999	1	0.999	
		Normal	1	0.06	0.112		0.994	0.999	0.996	
		Macro Average	0.744	0.58	0.57731		0.976	0.976	0.97583	
	C=10	DoS	0.814	0.675	0.738	78.1%	0.936	0.972	0.954	97.8%
		DDoS	0.757	0.863	0.807		0.977	0.928	0.952	
		Reconnaissance	0.857	0.96	0.9		0.983	0.988	0.986	
		Theft	0	0	0		0.999	1	0.999	
		Normal	1	0.095	0.174		0.994	1	0.997	
		Macro Average	0.685	0.519	0.524		0.978	0.978	0.97752	

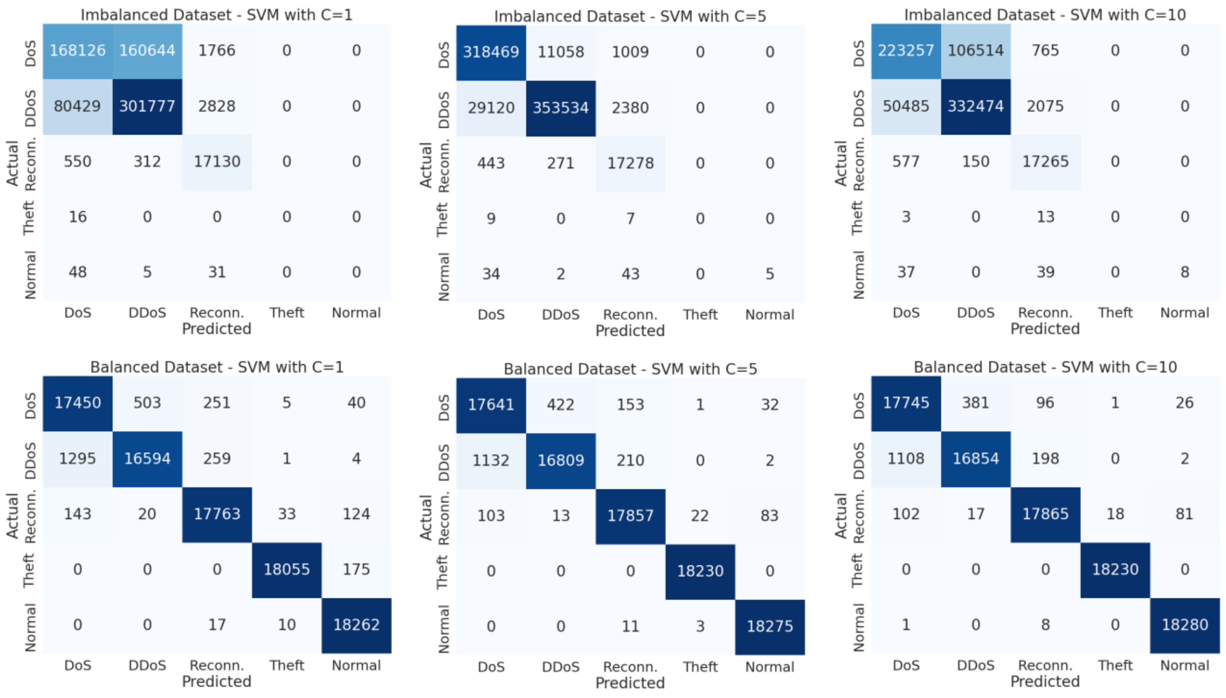


Figure 6: The confusion matrices of SVM models using balanced and imbalanced datasets.

5. Conclusion and Future Work

In conclusion, this work evaluates the impact of balancing the BoT-IoT dataset on the performance of learning-based IDSs using KNN, GB, and SVM. The study highlights the critical role of dataset balancing in enhancing the detection capabilities of these models. We employ both random under-sampling and SMOTE to address dataset imbalance. The findings of our experimental work demonstrate that balanced datasets reduce the disparity in class detection and significantly improve the models' performances, as reflected in increased F1-scores. The importance of this study lies in its practical implications for IoT security systems. It also demonstrates that accuracy alone can be misleading in imbalanced datasets and emphasizes the need to examine other metrics like precision, recall, and F1-score to ensure a more accurate and comprehensive evaluation. An effective IDS must accurately detect malicious activities while minimizing false positives and false negatives, which is crucial for maintaining the security and integrity of network systems.
Future work includes investigating the impact of different balancing methods such as Adaptive Synthetic (ADASYN). Another potential area of research is the application of deep learning techniques, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), to the balanced BoT-IoT dataset. Finally, real-time testing and evaluation of these models in a live network environment will be crucial to validate their effectiveness and reliability in IoT ecosystems.

References

1. D. Fawzy, S. Moussa and N. Badr, Data fusion for data prediction: an IoT-based data prediction approach for smart cities, IJICIS 23(2) (2023) 13-27.
2. D. K. A. A. Rizk, H. M. Hosny, S. ElHorbety and A. B. Salem, Smart hospital management systems based on Internet of Things: challenges, intelligent solutions and functional requirements, IJICIS 22(1) (2022) 32-43.

3. S. Helmy, A. Amar, and E. El-Horabty, Internet of Things (IoT) based smart device for cardiac patients monitoring using Blynk App, IJICIS 22(4) (2022) 86-99.
4. O. Elnakib, E. Shaaban, M. Mahmoud, and K. Emara, Federated learning enabled IDS for Internet of Things on non-IID data, IJICIS, 24(1) (2021), 13-28.
5. N. Aljehane, H. Mengash, S. Hassine, F. Alotaibi, A. Salama, and S. Abdelbagi, Optimizing intrusion detection using intelligent feature selection with machine learning model, Alex. Eng. J. 91(2024) 39-49.
6. S. Barkah, S. Selamat, Z. Abidin, and R. Wahyudi, Impact of data balancing and feature selection on machine learning-based network intrusion detection, JOIV 7(1) (2023) 241-248.
7. E. Ismail, W. Gad, and M. Hashem, SMOTE-RUS: Combined oversampling and undersampling technique to classify the imbalanced autism spectrum disorder dataset, IJICIS 23(3) (2023),83-94.
8. N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset, FGCS 100 (2019) 779-796.
9. M. Ayyagari, N. Kesswani, M. Kumar, and K. Kumar, "Intrusion detection techniques in network environment: a systematic review," WIREL NETW (2021) 1-17.
10. MIT Lincoln Laboratory, DARPA intrusion detection evaluation dataset, 1998. [Online]. Available: https://www.ll.mit.edu/r-d/datasets/1998-darpa-intrusion-detection-evaluation-dataset.
11. KDD Cup 1999 Data, [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.
12. NSL-KDD Canadian Institute for Cybersecurity, [Online]. Available: https://www.unb.ca/cic/datasets/nsl.html.
13. J. Song, H. Takakura, Y. Okabe, M. Eto, D. Inoue, and K. Nakao, Statistical analysis of honeypot data and building of Kyoto 2006+ dataset for NIDS evaluation, in Proceedings of the first workshop on building analysis datasets and gathering experience returns for security, 2011.
14. N. Moustafa and J. Slay, UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set), in 2015 Military Communications and Information Systems Conference (MilCIS).
15. CICIDS 2017, [Online]. Available: https://www.unb.ca/cic/datasets/ids-2017.html.
16. X. Gao, C. Shan, C. Hu, Z. Niu and Z. Liu, An Adaptive Ensemble Machine Learning Model for Intrusion Detection," IEEE Access, 7 (2019) 82512-21.
17. Z. Chkirbene, S. Eltanbouly, M. Bashendy, N. AlNaimi and A. Erbad, "Hybrid machine learning for network anomaly intrusion detection," In IEEE international conference on informatics, IoT, and enabling technologies (ICIoT), 2020, 163-170.
18. M. A. Ferrag, L. Maglaras, A. Ahmim, M. Derdour, and H. Janicke, Rdtids: Rules and decision tree-based intrusion detection system for internet-of-things networks, Future internet 12(3) (2020) 44.
19. S. Dina, A. Siddique, and D. Manivannan, Effect of balancing data using synthetic data on the performance of machine learning classifiers for intrusion detection in computer networks., IEEE Access, 10(2022) 96731-47.
20. M. Douiba, S. Benkirane, A. Guezzaz, and M. Azrour, An improved anomaly detection model for IoT security using decision tree and gradient boosting, J. Supercomput. 79(3) (2023) 3392-3411.
21. H. Xu, Z. Sun, Y. Cao, and H. Bilal, A data-driven approach for intrusion and anomaly detection using automated machine learning for the Internet of Things, Soft Comput. 27(19) (2023) 14469-81.

22. L. Sasse, E. Nicolaisen-Sobesky, J. Dukart, S. B. Eickhoff, M. Götz, S. Hamdan, V. Komeyer, A. Kulkarni, J. Lahnakoski, B. Love, and F. Raimondo, On Leakage in Machine Learning Pipelines, arXiv preprint arXiv:2311.04179, 2023.
23. N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique.," J. Artif. Intell. 16 (2002) 321-57.